Katrine Chow
CS 162 Winter 2018
Project 2 Design and Reflection

Project 2 – Design and Reflection Document

Problem:

Project 2 asks us to build a Zoo Tycoon game, which is a zoo management simulation. Player starts off with a starting bank amount, and can use the money to purchase various animals. Each animal has its own cost, feed, and payoff attributes. They may also reproduce offspring or die during random events.

Requirements:

a) Implement a mechanism to keep track of day cycle.
b) Player must purchase 3 different types of animals at the start of the game, up to 2 for each type.
c) During a day cycle, all animals increase age by 1 day.
d) Feed cost is collected and subtracted from bank amount.
e) A random event may or may not take place. Events include: Birth, Death, and Bonus reward.
f) At the end of the day, tally up the animals' payoff and add to bank total.
g) Ask player if they'd like to continue or quit.
h) Game quits when player has less than $0 in their bank.
i) Classes: zoo, animal (with derived classes tiger, penguin, and turtle).
j) Zoo class should contain dynamic arrays of each animal type. Starting with capacity of 10. If needed, arrays must expand by doubling its size.
k) Input Validation

Program Design:

See attached flowchart.

This program offers extensive practice of inheritance and polymorphism. Classes Tiger, Penguin, and Turtle must inherit member functions from class Animal. These classes mainly keep track of various attributes of each animal.

The Zoo class manages the daily tasks of running the zoo, including purchasing of animals, increases the animals' age, calculates feed costs, and generates random events.

A Bank class is added to handle all bank-related functions: deposits, withdrawals, and getting the total amount.

The Menu class is reused from previous assignments, and it is tailored to this program.

Test Plan:

See attached PDF.

Reflection:

I learned a lot from this Project, and I can see rapid improvement in how I solve problems. After not doing as well as I'd like on Langton's Ant, I was advised to focus on pen-and-paper design. I went through probably a dozen sheets for this Project and mapped everything out with actual code.

This method definitely saved me so much time and allowed me to better understand the flow of the program. I also completed Lab 4 before I started this Project, and that experience really helped me code and debug this game.

Since I had a detailed plan, I didn't deviate too much from my original design. I also learned a few more tricks of using gdb to debug, so I believe I sorted out all the seg faults this time around! (I hope).

I did struggle a bit with the dynamic arrays and memory management. It seems like I either didn't free up all the space, or freed them up too soon. I do think I have a better grasp on pointers and dynamic allocation after Lab 4 and this project. I also really appreciate the practice on inheritance. Having to use raw pointers forced me to think thoroughly on my design. I appreciate vectors even more after this project!

I had a lot of fun working on this project. Even though it did not turn out exactly the way I'd like, I think I am pretty proud of what I've made! This is only week four, and I am already seeing a significant improvement in how I think about code. I plan to study more on valgrind and dynamic memory, and hope to do better in my next project.